5/PRTS

AN INTERNET CACHING SYSTEM AND A METHOD AND AN ARRANGEMENT IN SUCH A SYSTEM

Technical field of the invention

The present invention refers to an Internet caching system and to an arrangement and a method for serving requests for Internet information files in an Internet caching system.

Background art

The Internet and its currently most used feature, the World Wide Web (WWW), has in recent years developed into an enormous source of information. Anybody can provide any information, such as text, pictures, audio and video, on the World Wide Web where it can be easily retrieved by users anywhere in the world as long as they have access to the Internet.

The major problem facing the Internet is the growing demand for communication capacity as users access information from anywhere in the world. It is estimated that the World Wide Web traffic already exceeds all conventional telephone and facsimile traffic on most international communication lines. More transmission and switching capacity is continuously added, but it is a slow and expensive process and demand continues to outstrip supply.

The content of the World Wide Web is getting to be unmeasurable and probably comprises several hundreds of Terabytes (as of summer 1998). However, a relatively small subset of all this information accounts for a huge portion of the information actually being accessed. Therefore, in order to minimize bandwidth used and latency involved when accessing information on the Internet, different caching techniques are currently in use for limiting the amount of information that has to be

transferred over the Internet and for limiting the distance over which the information is transferred.

In the field of caching WWW objects, or Internet information files, there are basically two approaches, client side caching and server side caching. The simplest form of client side caching is virtually used by every WWW browser today. The browser retains a cache on the user's computer with the last accessed Internet information files. When the user for a second time wishes to access a particular information file, the browser retrieves it from its cache rather than making a request for it over the Internet.

In order to help a neighboring user, a proxy server caching method, another form of client side caching, can be used. In this scheme a cache is placed at a WWW proxy node to which a number of neighboring users are connected, such a proxy node could for example be a server located at a company. When a WWW client wants to access a WWW server on the Internet, the client sends a http request to the proxy node, or WWW proxy server, rather than sending it directly to a server on the global Internet. Instead it is the proxy server that sends the request to a WWW server on the global Internet, caches the response and returns the response to the client. Thus, the first time an information file is requested it is transferred over the Internet and stored in the cache of the WWW proxy server. Subsequent requests for the same information file from any client connected to the WWW proxy server can then be resolved locally, rather than making http requests to a WWW server over the global Internet. Proxy server caching can also be used outside the premises of a company, or some other organization, by implementing the scheme described above at a regional Internet cache server to which a number of clients are directly or indirectly connected.

Depending on the size and homogeneity of a user community using a cache at a server, about 20-40 Giga-

bytes of cache storage will (spring 1998) reduce the Internet traffic generated by the user community by 30-50 %. As the growth of the information provided by the Internet and the WWW continues, it is highly likely that the required cache size will have to increase over time to retain the hit rate, i.e. the proportion of the information files requested that are transferred from the cache server. Furthermore, it would give significant benefit for the performance and utilization of the Internet if the hit rate could be increased to 75% or more. With the typical end user behavior, this would require a much larger cache, currently in the order of 200-400 Gigabytes, but also require very many members in the end user community, currently several hundreds of thousands. The reason is that the larger the end user community, the larger the probability that someone else within the community has previously accessed a requested file, especially if the users share some common interest.

Installing a large cache is easily achieved by acquiring the appropriate computer and the appropriate disk capacity. However, it is also required that the cache is able to handle all requests from the participating end users. Using current technology, it is not possible for one single processor computer to serve the requests from several hundred thousand end users. Hence, several systems have been presented to deal with this problem, here outlined under the names of their major proponents.

Cisco Systems, Inc. proposes that the end users are connected to a backbone router which is programmed to transparently redirect all WWW requests to a group, or "Farm" of dedicated cache appliances, or "Cache Engines". Each Cache Engine handles a subset of all origin WWW servers, based on grouping of the IP (Internet Protocol) addresses. The solution scales up to 32 Cache Engines in parallel, which corresponds to serving approximately 500.000 subscribing end users.

Inktomi Corporation suggests that a switch, a so called layer 4 switch, is used to redirect all requests for WWW pages to an "Inktomi Traffic server". A cluster of powerful computers are used, which all share the same disk storage system. This solution scales up to 16 parallel workstations, which also corresponds to about 500.000 subscribing end users. However, having several computers accessing the same disk storage system adds complexity and requires management, i.e. some of the capacity of each computer is not available for processing requests.

Network Appliance, Inc. proposes a two tier caching solution. The system has several local caches near the end users. These local caches communicate with a central cache using the Internet Cache Protocol (ICP) when a cache miss occurs at the local level. If the requested file is present in the central cache, it will be transferred to the local cache and then forwarded to the end user. If the requested file is not in the central cache either, the central cache will make a request to the origin server and forward the file to the local cache, which in turn forwards the file to the end user. The central cache thus handles ICP requests from the local caches and communicates with the origin server in the case of a cache miss at the central cache. For scaleability, there can be several central caches in parallel, each handling a subset of the origin servers. This means that the local caches are able to address each request to the correct central cache server. Since this protocol is not standardized, it means that all local caches have to be delivered from Network Appliance, Inc.

All of these solutions have the drawback that a central cache server needs to handle extensive communication in one way or another. This results in low utilization of the server's capacity and difficulties in serving hundreds of thousands users, which is required in order to obtain a high hit rate. By adding more servers,

the systems are made more expensive and more complex. The complexity of the system adds to the overhead and, hence, to a low utilization of the relatively expensive resources that the servers represent.

Summary of the invention

An object of the present invention is to overcome the drawbacks with the presently known techniques for caching information files on the Internet and to provide a solution for caching information files in a cost-effective way.

Another object of the present invention is to provide a solution for how user's requests for cached information files are to be served by a caching system in a fast and cost-effective way.

Yet another object is to provide a cache server solution which is able to cope with the growing numbers of information files being provided by the Internet and the World Wide Web.

Yet another object is to provide a solution for obtaining a high hit rate percentage for information file requests directed to a caching system with a minimum of cost.

Yet another object of the present invention is to provide a scaleable caching system which is scaleable in a standardized way.

The above objects are achieved by an Internet caching system and a method for serving requests for Internet information files in an Internet caching system in accordance with the appended claims.

According to a first aspect of the invention, there is provided a method for serving requests for Internet information files in an Internet caching system, which method comprising the steps of receiving, at a local Internet cache server, a user request from a user for an Internet information file; in response to the received request, making a query for said information file, if

said information file has not been cached by said local server; in response to a reply to said query, making a file request for said information file, wherein said file request is directed to a feeder means if said reply indicates that a central file server, storing cached Internet information files, has said information file cached; and querying, from said feeder means in response to said file request, said central file server for said information file, in order to decrease the load on said central file server.

According to a second aspect, there is provided an arrangement in an Internet caching system, said system comprising at least one local cache server and at least one central file server, both of which servers stores cached Internet information files, which arrangement, for decreasing the load on said central file server, includes a Feeder communicating with said local cache server and with said central file server, wherein said Feeder includes first means for receiving a request for an Internet information file from said local cache server; second means for deriving a query from an alphanumerical string received from said local cache server; and third means for querying said central file server for said Internet information file using said query derived by said second means.

According to a third aspect, there is provided an Internet caching system, which system comprises a set of local Internet cache servers, wherein each local cache server is arranged to receive requests from users for Internet information files; at least one central file server included in a central cache site and storing cached Internet information files; and feeder means interconnecting said set of local cache servers with said central file server, said feeder means including at least one Feeder, which Feeder comprises means for communicating with at least one local cache server in accordance with a protocol used for communicating between Internet

cache servers and means for retrieving Internet information files from said central file server using data base queries, thereby decreasing the load on said central file server.

The invention is based upon the idea of connecting a number of dedicated computers to a central file server, or central cache server, storing Internet information files. Relative to the central cache server, these additional computers are low end computers. The dedicated computers are arranged to decrease the load on the central cache server by performing some of the tasks normally handled by the central cache server itself. In this way the central cache server is able to serve the local cache servers connected to the central server, or rather connected to the central server via the dedicated computers, in a fast and cost-effective way. Maximum use is made of the expensive hardware forming the actual central file server and its file repository in which the files are cached, while specialized inexpensive machines around the file server perform time consuming and time critical tasks in parallel.

Thus, the inventive feeder means, or Feeders, are realized by machines being separate from any machine realizing a central file server. This will decrease the load on the central file server, which then is able to dedicate more processing time to the actual retrieval of cached information files. Hence, the central file server is able to serve a large community of users in an efficient way. Since user requests, via requesting local cache servers, are served more effectively, the number of user requests served can be increased, which in turn enables the central file server to obtain a higher hit rate percentage for its cache.

According to an embodiment of the present invention, the feeder means communicates with the local cache servers, on behalf of the central file server, in accordance with a protocol used for communicating between

Internet cache servers. The currently used protocol is either the Internet Cache Protocol (ICP) or the Cache Digest, but could be any other conventional or future protocol used for the same purpose. Thus, by placing the task of accepting, and replying to, queries and/or requests for information files in machines being separate from the central file server machine, the load on the central file server is decreased considerably.

When a local cache server receives a request from a user for an information file, which file has not been cached at the local server, the local server starts with making a query for that file. In one embodiment the query is directed to a table, or data base, being internal to, or directly connected to, the local server. If said table indicates that the queried file is cached by the central file server, the local server will request the file from the feeder means, or Feeder. This querying and requesting is then preferably performed in accordance with the Cache Digest protocol. However, as with the request from the user to the local server, the request from the local server to the Feeder may be in accordance with any layer three protocol, for example an HTTP request.

In another embodiment, the query from the local server is directed to the Feeder. Included in the query, for example an ICP query, is the URL of the queried information file. The Feeder derives a query number from the alphanumerical URL of the received query for an information file, which query number then is used by the Feeder for querying the central file server for the information file. The Feeder queries the file server for information files using a standard SQL query (Structured Query Language). If the queried file is present at the central file server, i.e. if there is a cache hit, the queried file is transferred from the central server, via the Feeder, to the local server. To have the central file server initiate a file transfer as an answer to a SQL query, rather than as an answer to a query, such as an

ICP query, from the local cache server, means considerable capacity savings at the central file server.

Alternatively, the query number is derived from said alphanumerical URL and from a part of a header information included in said query. This part of the header information contains specific user information of the original requester, for example, the language he is using, enabling the central file server to respond in accordance with this specific information. The query number corresponding to an information file is derived by using any hash algorithm, preferably using an MD5 hash algorithm.

In the embodiment where the local server makes an internal query for the information file, the Feeder derives the query number from the following request directed to the Feeder by the local server. The alphanumerical string used for deriving the query number is the string included in said request, for example the URL of an HTTP request. The query number is then used by the Feeder when querying the central file server for the information file, preferably using an SQL query. Again, it is advantageous to also include at least part of an header information field of said request as the basis for deriving said query number.

In order to decrease the load on the central file server even further, the Feeder preferably includes a table storing information relating to each information file being cached by the central file server. The table, for example, being a memory resident MD5 indexed hash table. By searching said table, the Feeder can conclude whether or not a queried information file is cached by the central file server, without having to query the server, and, hence, a faster reply may be given by the Feeder to the query from a local server.

According to another embodiment of the present invention, the Internet caching system further comprises updater means, or an Updater, for updating the set of

information files being cached by the central file server. The updating procedure consists of transferring a copy of a file cached at a local server to the central server. The transferred file is a file which, as a consequence of a cache miss at the central server when querying for the file, has been retrieved from its origin server by the local server and then been cached by the same.

Thus, the central file server, or central cache server, does not itself retrieve a non-cached file and is therefore not burden with having to make a request to an origin server for a file because of a cache miss when serving a local cache server. Instead, when the Feeder evaluates a query from the local cache server for an information file, and concludes that the queried file is not cached at the central file server, the Feeder directs a reply to the querying local server, indicating that the file is not available, and then orders the Updater to update the central file server. Upon reception of the reply, which thus indicates a cache miss, the local cache server retrieves the file in question from its origin server. Upon reception of the order to update the central file server, the Updater requests a copy of the file from the local server and transfers the thereby received file copy to the central cache server where it is stored. The transferring and storing procedure is preferably performed at a time when the overall load on the central file server is low, and when the local server has been given enough time to retrieve the file from its origin server.

However, should the local server be located behind a firewall, the Updater will request a copy of the file from its origin server, which copy then is stored in the central cache server. In this case, it is preferred that the Feeder does not order the Updater to commence the updating procedure until after a certain number of queries for the same particular information file have

been received, where these queries originate from local servers being located behind firewalls. Preferably, the Updater is realized by a machine being separate from the machines realizing the Feeders, as well as being separate from any file server machine. This is an advantage since file requests, for example HTTP requests, to origin servers may take unpredictable amounts of time and thus lead to an unpredictable load on the machine performing the requests. However, in a simplified system it is possible to realize the Updater in the same machines as those which realize the Feeders, while still being separate from any central file server machine. In an embodiment where the machines implementing the Updater and the Feeders interconnects the local cache server with the central file server, without the machines themselves being included in the central cache site together with the central file server, the separation of these machines from the central file server machine is evident.

Certain Internet information files are not suitable for caching. Such files are sometimes called dynamic information files, the term dynamic comes from that these files are continuously updated at the origin server, examples of such files are files with stock quotes, weather reports and so on. One preferred way of treating the existence of dynamic files is to uphold a list of known uncachable files in either the Updater or in the local servers. In this way the communication in the system, as a result of a user requesting such a file, can be minimized.

According to yet another embodiment of the present invention, several central file servers are included in a central cache site, each file server caching information files associated with original host names, IP-addresses or derived query numbers, within a defined range. Based upon either the original host name, IP-address or the derived query number of a requested information file, the Feeder addresses the query to the file server caching

files in the appropriate range. In this scaleable solution each file server has its own disk system, thus minimizing overhead. Furthermore, the central cache site is scaleable with third party file servers because of the standardized protocols used by the site.

In order for the communication between the central file server and the low end computers, i.e. the Feeders and the Updaters, to be fast, each low end computer is preferably connected to the central file server by means of a dedicated wire, alternatively, if there are several file servers, by means of a dedicated network. This network is either a private or a public network. In the latter case, at least part of the network capacity is preferably reserved for the communication in question. The network used can, of course, also be a part of the Internet, also in a non-dedicated way. The type of connection used between the central file server and the low end computers is very much dependent upon where the low end computers, or Feeders and Updaters, are located, at the same site as the central file server, or, at a location being different from the location of the central file server.

Moreover, it is preferred that the central cache site serves a defined set of local cache servers, which set in turn serves a linguistically and culturally homogenous user community. This will further increase the hit rate percentage at the central cache level since it is more likely that the same information files are requested more than once.

Using the present invention, an operator of an Internet caching system, handling information file requests in accordance with the present invention, is able to provide a fast, cheap and effective way of serving a large number of subscribing customers. These customers preferably being different Internet service providers, companies or other organizations connected to the inventive central cache site, or inventive Feeders/

Updaters, with their own local cache servers, or, being connected as clients to a system encompassing the whole inventive caching system formed by the central cache site, including Feeders and an Updater, and its connected local cache servers. Of course, a customer may very well also be a single user constituting a single WWW client connected directly to the inventive system. Also, a large company or Internet service provider can choose to operate the inventive system on its own rather than being connected to such a system being operated by another party. Furthermore, since the inventive caching system is built around standardized protocols, such as ICP and SQL, local cache servers and central file servers from any manufacturer can be included in the system as long as these protocols are supported.

Within the scope of the present invention a local Internet cache server is to be interpreted as a proxy node, preferably a WWW proxy node, retaining a cache for the users, or WWW clients, connected to the proxy node.

Items cached on a local Internet cache server or a file server at a central cache site are any non-dynamic files which are accessible using the Internet and containing any type of information. Thus, a number of different type of files and different namings of such files are included by the term Internet information file used in the present invention, such as binary, text, picture, audio and video files, HTTP (HyperText Transfer Protocol) files, WWW files, FTP (File Transfer Protocol) files, WWW pages, WWW objects, and so on. Besides files being accessed using the HTTP or FTP protocol, any file being accessed over the Internet in accordance with any layer 3 protocol is also included by the term Internet information file. A further example of a protocol that can be used is the WTP protocol (Wireless Transport Protocol) used within the WAP (Wireless Application Protocol) standard.

According to a fourth aspect of the present invention, the invention encompasses a computer-readable medium, on which is stored one or several computer programs of instructions for one or several general purpose computers, comprising means for enabling said one or said several computers to perform the steps disclosed in the appended claims 1 - 17.

According to a fifth aspect of the present invention, the invention encompasses one or several program storage devices containing one or several sequences of instructions, for one or several general purpose computers, for performing the steps disclosed in the appended claims 1 - 17.

The above mentioned and further aspects and features of, as well as advantages with, the present invention, will be more fully understood from the following description, with reference to the accompanying drawings, of exemplifying embodiments thereof.

Brief description of the drawings

Exemplifying embodiments of the invention will now be described below with reference to the accompanying drawings, in which:

Fig. 1 schematically shows an embodiment of an Internet caching system according to the present invention;

Fig. 2 schematically shows another embodiment of an Internet caching system according to the present invention;

Fig. 3 schematically shows a flow chart of the operations performed by a local cache server in Fig. 2;

Fig. 4 schematically shows a flow chart of the operations performed by a Feeder in Fig. 2;

Fig. 5 schematically shows a flow chart of the operations performed by an Updater in Fig. 2; and

Fig. 6 schematically shows yet another embodiment of an Internet caching system according to the present invention.

Detailed description of preferred embodiments

With reference to the block diagram shown in Fig. 1, an embodiment of the present invention will be described. In Fig. 1 a number of local cache servers 100 are shown. These local servers 100 are, via the Internet, connected to feeder means 110, here exemplified with a Feeder 110. The number of Feeders 110 and the number of local cache servers 100 indicated in Fig. 1 is merely an example, and the embodiment is not restricted to these numbers.

However, regardless of the number of Feeders, each Feeder is in this embodiment connected to one single central file server. In Fig. 1 Feeder 110 is connected to a central file server 130. This central file server includes a storage medium (not shown) on which Internet information files are stored, i.e. cached, and is implemented by a high end computer, such as a Sun Ultra Sparc or DEC Alpha Computer. Each Feeder 110 on the other hand is implemented by a low end computer, such as a conventional Personal Computer, and constitutes a front end machine which handles the communication between the local cache servers 100 and the central file server 130.

The Feeder 110 communicates with the local cache servers 100 using the Internet Cache protocol, which is a message based protocol used for communicating between cache servers over the Internet. Hence, the Feeder 110 replies to an ICP query for a cached Internet information file, the query being received from one of the local cache servers 100, with an ICP reply. This ICP reply indicates either a cache hit (ICP_OP_HIT) or a cache miss (ICP_OP_MISS).

In accordance with the Internet Cache Protocol, the ICP query received by the Feeder includes the URL of the queried information file. From this URL the Feeder 110

derives a query number, corresponding to the queried information file, using an MD5 hash algorithm. Using the query number, a memory resident MD5 indexed hash table 115 is then searched. Included in the Feeder 110 is a RAM (Random Access Memory) 116 in which the indexed table is stored. The indexed table 115 comprises an entry for each query number corresponding to an Internet information file cached at the central file server 130. Searching the indexed table 115 comprises searching the entries for a query number matching the derived query number. If a matching query number is found in the table, this is an indication that the queried information file is cached by the central file server 130, and, as a consequence, the ICP rely to the local server 100 will indicate a cache hit. Correspondingly, if a matching query number is not found in the table 115, this indicates that the queried information file is not cached by the central file server 130, and, as a consequence, the ICP reply will indicate a cache miss.

The means for deriving the query number using the MD5 hash algorithm and for searching the indexed table is a microprocessor 120, together with an appropriate software module, included in the Feeder 110. The microprocessor executes the software module, which execution results in derived query number and in a search of the indexed table 115. The implementation of this software module is straight forward for a person skilled in the art of programming.

If the reply from the Feeder 110 to the local server 100 indicates a cache hit, the local server will request the information file from the Feeder using the HyperText Transfer Protocol, which is a protocol used for accessing WWW objects over the Internet. That is, an HTTP request is transmitted to the Feeder, which request includes the URL of the requested file.

When communicating with the central file server 130, the Feeder 110 uses common SQL queries. Upon reception of

the HTTP request, the Feeder will retrieve the query number which was previously derived from the URL of the corresponding ICP query. Alternatively, the URL of the HTTP request is used for once again deriving the query number. The Feeder then uses the query number in a standard SQL query directed to the central file server. As a response, the central file server 130 will transfer the information file in question to the Feeder 110, which in turn transfers the information file to the local server 100 that issued the request for the file.

If the reply from the Feeder 110 to the local server 100 indicates a cache miss, the local server will make an HTTP request to the origin server (not shown) of the requested file, cache the then received file and transfer a copy of the file to the requesting user (not shown).

The means for implementing the execution of the Internet Cache Protocol in the Feeder 110 is the microprocessor 120 included in the Feeder. The microprocessor also implements the means for receiving an HTTP request from the local server 100 as well as the means for querying the central file server 130 using the SQL. The operations to be performed by the microprocessor are controlled by appropriate software modules, being part of the abovementioned means. The implementation of these software modules would be well known for a person skilled in the art of programming and being familiar with the protocols in question.

Another embodiment of an Internet caching system according to the invention is described with reference to Fig. 2. The system in Fig. 2 differs from the one shown in Fig. 1 in that the Internet caching system comprises an Updater 240, i.e. updater means, being connected to the central file server 230, the Feeder 210 and, via the Internet, the local cache servers 200. Thus, Fig. 2 illustrates the inventive arrangement encompassing an Updater 240 as well as a Feeder 210.

Besides what is being described below regarding the elements in Fig. 2, the elements of Fig. 2, having corresponding elements in Fig. 1, operate and interact in accordance with what has previously been described with reference to Fig. 1. Therefore, only the features of these elements being of relevance to the embodiment illustrated by Fig. 2 are described below.

The Updater 240 is responsible for updating the storage medium (not shown) associated with the central file server 230 with new cached information files. As described with reference to Fig. 1, when the local server 200 receives a cache miss in an ICP reply from the Feeder 210, as a response to a previous ICP query to the same, the local server 200 makes an HTTP request for the file to its origin server (not shown). The requested file is then received and cached by the local server 200. After a predetermined time, as a consequence to the reported cache miss in the ICP reply, the Feeder 210 will instruct the Updater 240 to update the central file server.

The Updater 240 receives, from the Feeder 210, the URL of the queried file and the identity of the local server 200 which queried for the file. An HTTP request for the file is then made from the Updater to the specific local server. Upon reception of the requested file, the Updater stores, i.e. caches, the file at the central file server 230. When the file has been stored, the Updater instructs the Feeder to add the query number corresponding to the file in question in the indexed table 215 stored in the RAM area 216.

The means for requesting the information file from the local cache server 200 and the means for caching the received information file at the central file server 230 is a microprocessor 260, together with appropriate software modules, included in the Updater 240. The implementation of these software modules would be well known for a person skilled in the art of programming.

An example of the operations performed by a local cache server 200 in the embodiment of Fig. 2 will now be described with reference to the flow chart in Fig 3.

In step 300 the local cache server 200 receives a request for an Internet information file from a client served by the particular local cache server. However, the file request may also be received from the Updater 240, which operates in accordance with the description referring to Fig. 5. The local cache server then in step 301 searches its locally cached files for the file requested. If it finds the file, the file is transferred to the requesting client or to the Updater 240, this is indicated with step 302.

If the local cache server 200 does not find the requested file during the search, i.e. it has not cached the requested file, it examines in step 303 if the request came from the Updater. If this condition is true, a message is returned to the Updater in step 304 indicating that the requested file is not available. If the conditional step 303 is false, i.e. if the request came from a client, an ICP query is sent in step 305 to the Feeder 210. In the next step 306, the local cache receives an ICP reply from the Feeder 210 indicating whether or not the central file server 230 has the requested file cached. In step 307 the ICP reply is evaluated. If the reply indicates a cache miss, i.e. the requested file was not centrally cached, the local cache server 200 makes a HTTP request for the file directed to the origin server of the file. If the reply on the other hand indicates a cache hit, the local cache makes a HTTP request to the Feeder 210 for the file, this is indicated with step 309. In the next step 310, the local cache server receives the requested file from the Feeder. Finally, in step 311, the file is transferred to the client which requested the file.

The operations performed by the Feeder 200 in the embodiment of Fig. 2 is now described with reference to the flow chart in Fig. 4.

In step 400 the Feeder 210 receives an ICP query regarding an Internet information file from any of the local cache servers 200 being handled by the Feeder. The query includes the URL of the queried information file. From this URL the Feeder 210 in step 401 derives a query number using an MD5 hash algorithm, which query number is used in step 402 when searching an indexed MD5 hash table being resident in the memory 216 of the Feeder 210.

If the number is not found during the search in the hash table, the Feeder in step 403 sends an ICP reply indicating a cache miss back to the local cache server 200 from which the ICP query was received. In step 404 the Feeder 210 then orders the Updater 240 to retrieve the non-cached queried file by passing the URL of the queried file to the Updater. In step 405 the Feeder 210 adds the query number corresponding to the queried file in the indexed hash table 215. This is done in response to that the Updater 240 indicates to the Feeder that the queried file has been transferred from the local server 200 and stored in the central file server 230. The operation of the Updater 240 will be further described with reference to Fig. 5.

If the Feeder 210 in the conditional step 402 finds the query number during the search in the hash table 215, it will in step 406 send an ICP reply indicating a cache hit back to the local cache server 200 from which the ICP query was received. In step 407 the Feeder then receives an HTTP request from the local cache server 200 which previously issued the ICP query. Similar to the ICP query, the HTTP request includes the URL of the requested information file. In step 408 the Feeder 210 retrieves the previously derived query number corresponding to the file. With this query number the Feeder in step 409 queries the central file server 230 for the requested

information file using a standard SQL query. In step 410 the Feeder as a response receives the cached information file from the central file server 230, and in the next step 411, the requested cached Internet information file is transferred from the Feeder 210 to the requesting local cache server 200.

The operations performed by the Updater 240 in the embodiment of Fig. 2 is now described with reference to the flow chart in Fig. 5.

In step 500, the Updater 240 receives an order from the Feeder 210 indicating that a particular file should be requested. The file in question was previously requested by the local cache server 200, but the Feeder found that the central cache server 230 had not cached the file. The order includes the URL of the file as well as the address of the local cache server 200 which requested the file from the central cache 230. The Updater will then, in step 501, check the requested file of the order against a list of known uncachable files. If the list contains the requested file, the order will be discarded. If the list does not contain the requested file, the order is put on a hold by the Updater 240 so that there will be time for the local cache server 200 to retrieve the file from its origin server.

At a time convenient to the central file server 230, i.e. at a time with relative low load on the central server, the central server sends a message to the Updater 240 stating that any pending order should be executed, the reception of this message at the Updater 240 is indicated with step 502. In the next step 503, the execution of the order starts with that the Updater requests a copy of the file, which now should have been retrieved and cached locally, from the local cache server 200 from which the file request originated. A copy of the file is then received from the local cache server in step 504. In step 505, the received file copy is transferred to the central file server 230 to be cached by the same. In the

final step 506, the Updater 240 instructs the Feeder 210 to add the query number corresponding to the file cached at the central file server 230 to the indexed hash table 215.

The operation of the central file server 230 is straight forward. Basically it does two things, it answers SQL queries from the Feeders 210 by transferring cached files to them, and, it stores new information files in its cache, which files are transferred to it from the Updater 240.

Another exemplifying embodiment of an Internet caching system according to the present invention is now described with reference to Fig 6. In Fig 6, the system differs from the one shown in Fig 2 in that the system has more than one central file server, here being exemplified with three central cache servers 630. Also, Fig. 6 includes two Feeders 610, each of which is connected to its own set of local cache servers 600. The Feeders 610 and the Updater 640 are arranged together with the central file servers 630 at a central cache site 690. By means of an Ethernet network 680 arranged within the central cache site, the Updater 640 and each Feeder 610 are connected to all central file servers 630.

The additional number of central file servers in this embodiment enables more files to be cached and even more SQL queries to be answered by the central file servers in comparison with the embodiment of Fig. 2. Since the system is completely scaleable, any number of Feeders, Updaters or central file servers can in theory be added to the system.

The basic difference of the operation of the system in Fig. 6 to that of the system in Fig. 2 is that a Feeder 610 need to select one server, out of the plurality of central file servers 630, to which an SQL query should be directed. Each central file server 630 caches information files within original host names within a predefined range. Therefore, the selection of one of the

central file servers is done in accordance with the host name included in the URL received from the local server, either as part of an ICP query or as part of an HTTP request. When one of the central file servers has been selected by the Feeder, the SQL query with the derived query number is directed to that selected file server.

It is understood that the construction and function of the elements described with reference to the drawings will become apparent for those skilled in the art.

Even though the invention has been described with reference to specific exemplifying embodiments, many different alterations, modifications and the like will become apparent for those skilled in the art. The described embodiments are therefore not intended to limit the scope of the invention, as defined by the appended claims.